CLAIM AMENDMENTS

1. (currently amended) A method ~~of~~ for verifying the trustworthiness of ~~a~~ executable browser software, said method comprising the steps of:

transmitting via a distributed network an electronic document requiring a digital signature from a first user computer to a second user computer;

electronically signing the electronic document by the executable browser software at the second user computer to create a first digital signature;

including as an attribute of the first digital signature a second digital signature to verify the trustworthiness of the executable browser software itself, the second digital signature verifying the authenticity of at least one ~~or more~~ components running in an environment of the executable browser software ~~on the second user computer~~;

transmitting the signed electronic document from the second user computer to the first user computer; and

authenticating the second digital signature.

2. (currently amended) The method of claim 1, further comprising the step of determining whether the entity that executed the second digital signature is authorized to certify the trustworthiness of the at least one ~~or more~~ components.

3. (original) The method of claim 1, wherein the attribute is a signed attribute.

4. (original) The method of claim 1, wherein the attribute is an authenticated attribute.

5. (currently amended) The method of claim 1, wherein the authenticating step comprises verifying the authenticity of the second digital signature.

6. (original) The method of claim 5, wherein the authenticity of the second digital signature is verified using a digital certificate.

7. (currently amended) The method of claim 1, wherein the authenticating step comprises comparing a hash of the at least one ~~or more~~ components running in the executable browser software environment included in the second digital signature to a known-good hash of the at least one ~~or more~~ components running in the executable browser software environment.

8. (currently amended) The method of claim 1, wherein the authenticating step is performed by the first user computer.

9. (currently amended) The method of claim 1, wherein the authenticating step is performed by a computer maintained by a financial institution participant.

10. (currently amended) The method of claim 1, wherein the authenticating step is performed by an independent entity that is not a financial institution participant.

11. (currently amended) The method of claim 1, wherein the authenticating step is performed by the second user computer.

12. (currently amended) The method of claim 1, wherein an unsigned component running in the executable browser software environment of the second user computer is included as an attribute of the first digital signature.

13. (original) The method of claim 12, wherein the unsigned component is copied from RAM of the second user computer.

14. (original) The method of claim 12, wherein the unsigned component is copied from non-volatile memory of the second user computer.

15. (currently amended) The method of claim 1, wherein a hash of at least one ~~or more~~ signed executable browser software component~~s~~ running on the second user computer is included as an attribute of the first digital signature.

16. (currently amended) The method of claim 15, wherein the at least one ~~or more~~ signed component~~s~~ ~~are~~ is copied from RAM of the second user computer.

17. (currently amended) The method of claim 15, wherein the at least one ~~or more~~ signed component~~s~~ ~~are~~ is copied from non-volatile memory of the second user computer.

18. (currently amended) A method ~~of~~ for verifying the trustworthiness of ~~a~~ an executable browser software module, said method comprising the steps of:

creating a first set of known-good hashes, the first set of hashes comprising~~,~~:

a hash of the browser module at a first point in time, and

a plurality of hashes corresponding to a plurality of browser components at the first point in time~~,~~

~~wherein the first set of hashes comprise a known-good set of hashes~~;

determining the status of ~~a~~ the browser module ~~running on a computer~~ at a second point in time~~;~~ by creating a second set of hashes, the second set of hashes comprising~~,~~:

a hash of the browser module at the second point in time, and

a plurality of hashes corresponding to ~~a~~ the plurality of browser components ~~running on the computer~~ at the second point in time;

verifying the second set of hashes to ensure that each hash was created by a trusted source; and

subsequent to the executable browser software module having executed by virtue of having digitally signed an electronic document, comparing the first set of hashes to the second set of hashes to determine the trustworthiness of the browser module.

19. (currently amended) The method of claim 18, wherein the ~~step of determining is performed at a~~ second~~, subsequent~~ point in time is subsequent to the first point in time.

20. (currently amended) The method of claim 18, wherein the determining step ~~of determining~~ further comprises verifying the status of the browser module ~~if~~ when the first set of hashes matches the second set of hashes.

21. (currently amended) The method of claim 18, wherein the determining step ~~of determining~~ further comprises determining that the status of the browser module is bad ~~if~~ when the first set of hashes does not match the second set of hashes.

22. (currently amended) The method of claim 18, wherein the determining step ~~of determining~~ further comprises determining that the status of the browser module is unknown ~~if~~ when it ~~can~~ is not ~~be~~ determined that a hash in the second set of hashes was created by a trusted source.

23. (currently amended) The method of claim 18, wherein the determining step ~~of determining~~ further comprises determining that the status of the browser module is unknown ~~if~~ when it is determined that a hash in the second set of hashes was not created by a trusted source.

24. (currently amended) The method of claim 18, wherein the first set of hashes is maintained by a trusted entity, ~~and~~ said method further comprising the steps of:

receiving from a requestor a request to determine the trustworthiness of the browser module, the request including the second set of hashes;

generating a report about the status of the browser module based on a result of the determining step; and

transmitting the report to the requestor.

25. (original) The method of claim 24, wherein the steps of receiving, determining, generating, and transmitting are performed by the trusted entity.

26. (currently amended) The method of claim 18, wherein the second set of hashes comprises at least one ~~or more~~ hash~~es~~ of browser components at a second point in time.

27. (original) The method of claim 26, wherein the first set of hashes comprises hashes at a first point in time corresponding to the hashes in the second set of hashes.

28. (currently amended) The method of claim 27, wherein the ~~step of determining is performed at a~~ second~~, subsequent~~ point in time is subsequent to the first point in time.

29. (currently amended) The method of claim 27, wherein at least one ~~or more~~ of the hashes in the second set of hashes has been signed by a trusted source.

30. (currently amended) The method of claim 29, wherein the verifying step ~~of verifying~~ further comprises ~~for~~ verifying that a hash in the second set of hashes was created by a trusted source, by verifying the signature on the hash.

31. (currently amended) The method of claim 18, wherein the browser status request is received from a first customer seeking to verify the trustworthiness of ~~a~~ the browser module running on a computer in the possession of a second customer.

32. (currently amended) The method of claim 31, wherein ~~in~~ the first customer and the second customer are parties to a transaction.

33. (original) The method of claim 32, wherein the first customer is a buyer and the second customer is a seller in the transaction.

34. (original) The method of claim 31, wherein the second customer disaffirms the transaction based on the status of the browser.

35. (currently amended) ~~A system~~ Apparatus for providing trusted browser verification, said apparatus comprising:

a trusted verifier;

coupled to the trusted verifier, means for maintaining by the trusted verifier a first set of hashes generated by a microprocessor, the first set of hashes comprising a first hash of ~~a~~ an executable software browser~~,~~ and a first plurality of hashes corresponding to a plurality of browser components, the first set of hashes being a known-good set of hashes;

coupled to the trusted verifier, means for receiving by the trusted verifier a browser status request, the browser status request including a second set of hashes generated by a microprocessor, the second set of hashes comprising a second hash of the browser~~,~~ and a second plurality of hashes corresponding to a plurality of browser components ~~running on a microprocessor at a point in time~~;

coupled to the trusted verifier, means for verifying by the trusted verifier that each hash in the second set of hashes was created by a trusted source; and

coupled to the trusted verifier, means for determining by the trusted verifier the status of the browser, subsequent to the browser having executed by virtue of having digitally signed an electronic document, based on the first set of hashes and the second set of hashes.

36. (currently amended) The ~~system~~ apparatus of claim 35, wherein the trusted verifier determines the status of the browser by comparing the first set of hashes with the second set of hashes.

37. (currently amended) The ~~system~~ apparatus of claim 36, wherein the trusted verifier verifies the status of the browser ~~if~~ when the first set of hashes matches the second set of hashes.

38. (currently amended) The ~~system~~ apparatus of claim 36, wherein the means for determining determines that the status of the browser is bad ~~if~~ when the first set of hashes does not match the second set of hashes.

39. (currently amended) The ~~system~~ apparatus of claim 36, wherein the means ~~of~~ for determining determines that the status of the browser is unknown ~~if~~ when it ~~can~~ is not ~~be~~ determined that a hash in the second set of hashes was created by a trusted source.

40. (currently amended) The ~~system~~ apparatus of claim 36, wherein the means ~~of~~ for determining determines that the status of the browser is unknown ~~if~~ when it is determined that a hash in the second set of hashes was not created by a trusted source.

41. (currently amended) The ~~system~~ apparatus of claim 35, wherein the second set of hashes comprises at least one ~~or more~~ hash~~es~~ of browser components at a second point in time.

42. (currently amended) The ~~system~~ apparatus of claim 41, wherein the first set of hashes comprises hashes at a first point in time corresponding to the hashes in the second set of hashes.

43. (currently amended) The ~~system~~ apparatus of claim 42, wherein the ~~step of~~ determining means is ~~performed~~ invoked at a second~~, subsequent~~ point in time subsequent to the first point in time.

44. (currently amended) The ~~system~~ apparatus of claim 42, wherein at least one ~~or more~~ of the hashes in the second set of hashes has been signed by a trusted source.

45. (currently amended) The ~~system~~ apparatus of claim 44, wherein the means for verifying verifies that a hash in the second set of hashes was created by a trusted source, by verifying the signature on the hash.

46. (currently amended) The ~~system~~ apparatus of claim 35, wherein the browser status request is received from a first customer seeking to verify the trustworthiness of a browser running on a computer in the possession of a second customer.

47. (currently amended) The ~~system~~ apparatus of claim 46, wherein ~~in~~ the first customer and the second customer are parties to a transaction.

48. (currently amended) The ~~system~~ apparatus of claim 47, wherein the first customer is a buyer and the second customer is a seller in the transaction.

49. (currently amended) The ~~method~~ apparatus of claim 46, wherein the second customer disaffirms the transaction based on the status of the browser.

50. (currently amended) In ~~a system~~ conjunction with apparatus comprising a root entity, a first participant, a second participant, a first customer of the first participant, and a second customer of the second participant, a method for verifying the trustworthiness of a browser in possession of the first customer, said method comprising:

a) maintaining at a trusted verifier a first set of hashes, the first set of hashes comprising a first hash of the ~~first customer's~~ browser;

b) generating by the first customer a second set of hashes, the second set of hashes comprising a second hash of the ~~first customer's~~ browser;

c) transmitting by the first customer the second set of hashes to the second customer, using a network connection;

d) generating by the second customer a browser status request, the browser status request including the second set of hashes;

e) transmitting by the second customer the browser status request to the second participant;

f) forwarding by the second participant the browser status request to the trusted verifier;

g) determining by the trusted verifier a status of the ~~first customer's~~ browser;

h) generating by the trusted verifier a browser status response;

i) forwarding by the trusted verifier the browser status response to the second participant; and

j) transmitting by the second participant the browser status response to the second customer.

51. (original) The method of claim 50, wherein the trusted verifier determines the status of the browser by comparing the first set of hashes with the second set of hashes.

52. (currently amended) The method of claim 51, wherein the status of the browser is verified ~~if~~ when the first set of hashes matches the second set of hashes.

53. (original) The method of claim 51, wherein the status of the browser is one of good, bad, or unknown.

54. (original) The method of claim 50, wherein the trusted verifier verifies that each hash in the second set of hashes was created by a trusted source.

55. (currently amended) The method of claim 54, wherein the status of the browser is verified ~~if~~ when the first set of hashes matches the second set of hashes.

56. (original) The method of claim 54, wherein the status of the browser is one of good, bad, or unknown.

57. (currently amended) The method of claim 50, wherein ~~in~~ the first customer and the second customer are parties to a transaction.

58. (original) The method of claim 57, wherein the first customer is a buyer and the second customer is a seller in the transaction.

59. (original) The method of claim 58, wherein the second customer disaffirms the transaction based on the status of the browser.

60. (original) The method of claim 50, wherein the root entity establishes a set of operating rules for the system.

61. (original) The method of claim 50, wherein the first participant is a financial institution.

62. (original) The method of claim 50, wherein the second participant is a financial institution.

63. (original) The method of claim 50, wherein the first participant comprises a transaction coordinator for processing browser status requests.

64. (original) The method of claim 50, wherein the second participant comprises a transaction coordinator for processing browser status requests.

65. (original) The method of claim 50, wherein the trusted verifier is an integrated component of the first participant.

66. (original) The method of claim 50, wherein the trusted verifier is an integrated component of the second participant.

67. (original) The method of claim 50, wherein the trusted verifier is a distinct entity from the first and second participants.

68. (currently amended) ~~A system~~ Apparatus for verifying the trustworthiness of ~~a~~ an executable software browser in possession of a first customer, said apparatus comprising:

a root entity;

a first participant;

a second participant;

~~the~~ a first customer of the first participant;

a second customer of the second participant;

means for maintaining at a trusted verifier a first set of hashes, the first set of hashes comprising a first hash of the ~~first customer's~~ browser;

means for generating by the first customer a second set of hashes, the second set of hashes comprising a second hash of the ~~first customer's~~ browser;

means for transmitting by the first customer the second set of hashes to the second customer using a network connection;

means for generating by the second customer a browser status request, the browser status request including the second set of hashes;

means for transmitting by the second customer the browser status request to the second participant;

means for forwarding by the second participant the browser status request to the trusted verifier;

means for determining by the trusted verifier a status of the first customer's browser;

means for generating by the trusted verifier a browser status response;

means for forwarding by the trusted verifier the browser status response to the second participant; and

means for transmitting by the second participant the browser status response to the second customer.

69. (currently amended) The ~~system~~ apparatus of claim 68, wherein the trusted verifier determines the status of the browser by comparing the first set of hashes with the second set of hashes.

70. (currently amended) The ~~system~~ apparatus of claim 69, wherein the status of the browser is verified ~~if~~ when the first set of hashes matches the second set of hashes.

71. (currently amended) The ~~system~~ apparatus of claim 69, wherein the status of the browser is one of good, bad, or unknown.

72. (currently amended) The ~~system~~ apparatus of claim 68, wherein the trusted verifier verifies that each hash in the second set of hashes was created by a trusted source.

73. (currently amended) The ~~system~~ apparatus of claim 72, wherein the trusted verifier determines the status of the browser by comparing the first set of hashes with the second set of hashes.

74. (currently amended) The ~~system~~ apparatus of claim 73, wherein the status of the browser is verified ~~if~~ when the first set of hashes matches the second set of hashes.

75. (currently amended) The ~~system~~ apparatus of claim 74, wherein the status of the browser is one of good, bad, or unknown.

76. (currently amended) The ~~system~~ apparatus of claim 68, wherein ~~in~~ the first customer and the second customer are parties to a transaction.

77. (currently amended) The ~~system~~ apparatus of claim 68, wherein the first customer is a buyer and the second customer is a seller in the transaction.

78. (currently amended) The ~~system~~ apparatus of claim 68, wherein the second customer disaffirms the transaction based on the status of the browser.

79. (currently amended) The ~~system~~ apparatus of claim 68, wherein the root entity establishes a set of operating rules for the system.

80. (currently amended) The ~~system~~ apparatus of claim 68, wherein the first participant is a financial institution.

81. (currently amended) The ~~system~~ apparatus of claim 68, wherein the second participant is a financial institution.

82. (currently amended) The ~~system~~ apparatus of claim 68, wherein the first participant comprises a transaction coordinator for processing browser status requests.

83. (currently amended) The ~~system~~ apparatus of claim 68, wherein the second participant comprises a transaction coordinator for processing browser status requests.

84. (currently amended) The ~~system~~ apparatus of claim 68, wherein the trusted verifier is an integrated component of the first participant.

85. (currently amended) The ~~system~~ apparatus of claim 68, wherein the trusted verifier is an integrated component of the second participant.

86. (currently amended) The ~~system~~ apparatus of claim 68, wherein the trusted verifier is a distinct entity from the first and second participants.